**SUMMARY**

## USC ID/s

6074572947

## Datapoints

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| M+N | Time in MS  (Basic) | Time in MS  (Efficient) | Memory in KB (Basic) | Memory in KB (Efficient) |
| 16 | 0.0002 | 0.0004 | 16 | 32 |
| 64 | 0.0009 | 0.0079 | 64 | 80 |
| 128 | 0.0028 | 0.0698 | 192 | 112 |
| 256 | 0.0112 | 0.3239 | 848 | 416 |
| 384 | 0.0236 | 0.7573 | 1280 | 96 |
| 512 | 0.0412 | 1.5656 | 1600 | 336 |
| 768 | 0.0921 | 2.0021 | 3024 | 400 |
| 1024 | 0.1654 | 2.2367 | 5168 | 144 |
| 1280 | 0.2828 | 2.4869 | 7744 | 160 |
| 1536 | 0.3954 | 2.8621 | 11456 | 384 |
| 2048 | 0.7589 | 3.8460 | 19328 | 128 |
| 2560 | 1.1252 | 5.1345 | 21136 | 400 |
| 3072 | 1.7095 | 6.5633 | 31248 | 272 |
| 3584 | 2.0518 | 8.2935 | 34176 | 240 |
| 3968 | 2.7316 | 9.7231 | 40352 | 256 |

## Insights

### Graph1 – Memory vs Problem Size (M+N)

Chart, line chart

Description automatically generated

#### Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)

Basic: Polynomial

Efficient: Linear

#### Explanation:

Graph1 shows the performance of the efficient algorithm is improved compared to basic algorithm in terms of memory usage. Although the basic algorithm uses less memory for very short inputs (M+N < 64), the space complexity of the algorithm increases proportional to the input size, thus growing in polynomial space complexity (it is almost quadratic, however we can’t be sure because of the outliers and because of the limited number of inputs). On the other hand, the efficient algorithm shows a linear trend in which memory usage do not increase with increased input size. It must be noted that the trend of the efficient algorithm are not perfectly linear (it can be interpreted as polynomial as well).

### Graph2 – Time vs Problem Size (M+N)

Chart, line chart

Description automatically generated

#### Nature of the Graph (Logarithmic/ Linear/ Polynomial/ Exponential)

Basic: Logarithmic

Efficient: Polynomial

#### Explanation:

The runtime performance of the basic algorithm is better compared to efficient algorithm. In Graph2, it can be seen that the efficient algorithm grows almost *exponentially* up to some point (when the input size *M+N* is equal to 512), then switches to a *loglinear* trend (*nlogn*). Basic algorithm also shows *loglinear* trend for all its datapoints, hence we can say that it has a *logarithmic* time complexity. The time complexity of the basic algorithm grows slower compared to the efficient algorithm.

## Contribution

6074572947 : The project was completed individually.